Alexandria University
Faculty of Engineering
Computer and Systems Engineering
First Year

CS 111: Probability Theory and its Applications

Lab 3

Overview

In this lab, you will discover permutations and combinations operations you have studied in class using R and you'll be exposed to R statistical capabilities by studying various built in functions designed specially for statistical distributions.

Permutations and Combinations

Permutations with repetition

The number of permutations with repetition (or with replacement) is simply calculated by:

$$n^r$$

where n is the number of objects to choose from and r is the number of times.

Example
For example, you have an urn with a red, blue and black ball. If you choose two balls with replacement/repetition, there are 3^2 *permutations: {red, red}, {red, blue}, {red, black}, {blue, red}, {blue, blue}, {blue, black}, {black, red}, {black, blue}, and {black, black}. In R:*

```
install.packages('gtools')
#load library
library(gtools)
#urn with 3 balls
x <- c('red', 'blue', 'black')
```

```
#pick 2 balls from the urn with replacement
#get all permutations
permutations(n=3,r=2,v=x,repeats.allowed=T)
#      [,1]    [,2]
# [1,] "black" "black"
# [2,] "black" "blue"
# [3,] "black" "red"
# [4,] "blue"  "black"
# [5,] "blue"  "blue"
# [6,] "blue"  "red"
# [7,] "red"   "black"
# [8,] "red"   "blue"
# [9,] "red"   "red"
#number of permutations
nrow(permutations(n=3,r=2,v=x,repeats.allowed=T))
#[1] 9
```

Permutations without repetition

Calculating permutations without repetition (replacement) means that for cases where r > 1, n gets smaller after each pick where the order of balls is preserved. It is computed using the relation

$$^{n}P_{r}$$

where n is the number of objects to choose from and r is the number of times.

Example
For example, if we choose two balls from the urn with the red, blue and black ball but without repetition/replacement, the first pick has 3 choices and the second pick has 2 choices: {red, blue}, {red, black}, {blue, red}, {blue, black}, {black, red} and {black, blue}. In R:

```
#install if necessary
install.packages('gtools')
#load library
```

```
library(gtools)
#urn with 3 balls
x <- c('red', 'blue', 'black')
#pick 2 balls from the urn with replacement
#get all permutations
permutations(n=3,r=2,v=x)
#     [,1]    [,2]
#[1,] "black" "blue"
#[2,] "black" "red"
#[3,] "blue"  "black"
#[4,] "blue"  "red"
#[5,] "red"   "black"
#[6,] "red"   "blue"
#number of permutations
nrow(permutations(n=3,r=2,v=x))
#[1] 6
```

Combinations with repetition

The number of combinations with repetition (or with replacement) is simply calculated by:

$$S(n, r) = {}^{n+r-1}C_r$$

where n is the number of objects to choose from and you choose r from them.

Example :

for the same program used in the permutations case, the combinations without replacement is calculated in R using

```
    combinations(n, r, v=1:n, set=TRUE, repeats.allowed=True)
```

Combinations without repetition

Calculating combinations without repetition (replacement) means that for cases where r > 1, n gets smaller after each pick where we choose from a set of identical items. It is computed using the relation

$$^{n}C_{r}$$

where n is the number of objects to choose from and r is the number of times.

Example :

for the same program used in the permutations case, the combinations without replacement is calculated in R using

```
combinations(n, r, v=1:n, set=TRUE, repeats.allowed=FALSE)
```

Statistical Introduction

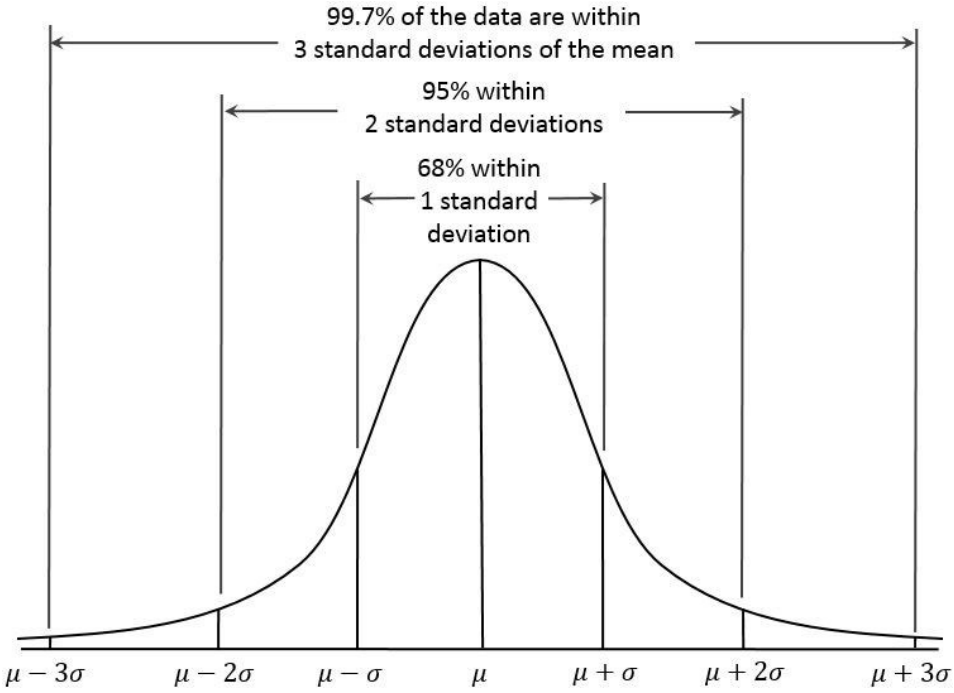
Basic Notations:

- **Mean** of a vector is the sum of values over the vector size. It is denoted by □ or □
- **Median** is the number in the middle of a vector of values (take care: even/odd cases)
- **Mode** is the value that has the maximum number of occurrences.
- **Variance** measures how far a set of (random) numbers are spread out from their average value. It is Often represented by **Var(x)** or ^{2} or s^{2}
- **Standard deviation** is a measure of the amount of variation or dispersion of a set of values.The standard deviation of a random variable, statistical population, data set, or probability distribution is the square root of its variance and is denoted by or s

Normal distribution:

Normal distributions are important in statistics and are often used in the natural and social sciences to represent real-valued random variables whose distributions are not known. In a random collection of data from independent sources, it is generally observed that the distribution of data is normal. Which means, on plotting a graph with the value of the variable in the horizontal axis and the count of the values in the vertical axis we get a bell shaped curve. The center of the curve represents the mean of the data set. In the graph, fifty percent of values

lie to the left of the mean and the other fifty percent lie to the right of the graph. This is referred as normal distribution in statistics.



R has four in built functions to generate normal distribution. They are described below.

```
dnorm(x, mean, sd)
pnorm(x, mean, sd)
qnorm(p, mean, sd)
rnorm(n, mean, sd)
```

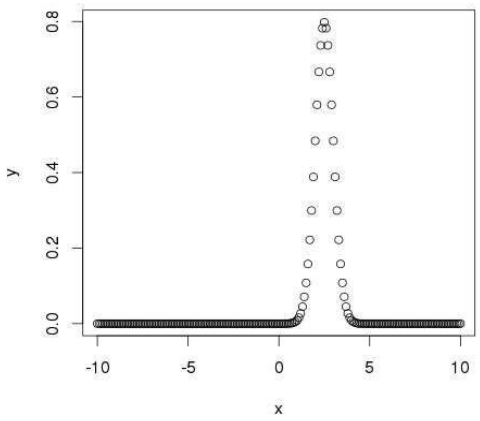
Following is the description of the parameters used in the above functions:

- x is a vector of numbers.
- p is a vector of probabilities.
- n is number of observations(sample size).
- mean is the mean value of the sample data. It's default value is zero.
- SD is the standard deviation. It's default value is 1.

1. dnorm()

This function gives the height of the probability distribution at each point for a given mean and standard deviation.

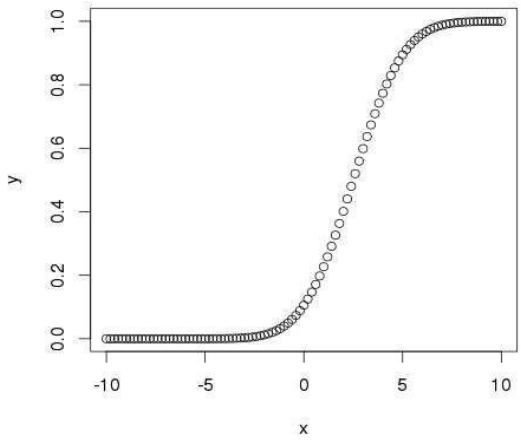
```
# Create a sequence of numbers
between -10 and 10 incrementing by
0.1.
x <- seq(-10, 10, by = .1)
# Choose the mean as 2.5 and
standard deviation as 0.5.
y <- dnorm(x, mean = 2.5, sd =
0.5)
# Give the chart file a name.
png(file = "dnorm.png")
plot(x,y)
# Save the file.
dev.off()
```



2. pnorm()

This function gives the probability of a normally distributed random number to be less than the value of a given number. It is also called "Cumulative Distribution Function"

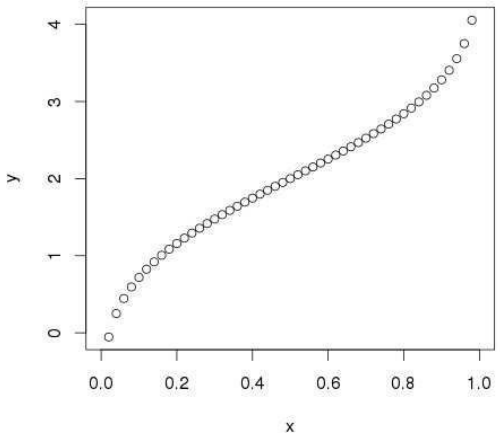
```
# Create a sequence of numbers
between
 -10 and 10 incrementing by 0.2.
x <- seq(-10,10,by = .2)
# Choose the mean as 2.5 and
standard
deviation as 2.
y <- pnorm(x, mean = 2.5, sd = 2)
# Give the chart file a name.
png(file = "pnorm.png")
# Plot the graph.
plot(x,y)
# Save the file.
dev.off()
```



3. qnorm()

This function takes the probability value and gives a number whose cumulative value matches the probability value.

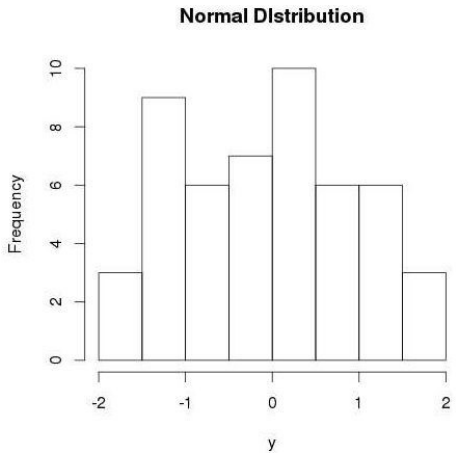
```
# Create a sequence of
probability
values incrementing by 0.02.
x <- seq(0, 1, by = 0.02)
# Choose the mean as 2 and
standard deviation as 3.
y <- qnorm(x, mean = 2, sd = 1)
# Give the chart file a name.
png(file = "qnorm.png")
# Plot the graph.
plot(x,y)
# Save the file.
dev.off()
```



4. rnorm()

This function is used to generate random numbers whose distribution is normal. It takes the sample size as input and generates that many random numbers. A histogram is drawn to show the distribution of the generated numbers.

```
# Create a sample of 50 numbers
 which are normally distributed.
y <- rnorm(50)
# Give the chart file a name.
png(file = "rnorm.png")
# Plot the histogram for this
sample.
hist(y, main = "Normal
DIstribution")
# Save the file.
dev.off()
```



Hints:

`png(file = "name.png")` *is used to rename the distribution plot*
`plot(x,y)` *is used to plot the distribution*
`dev.off()` *is used to save the file*
`nrow()` *counts the number of rows*

Application

You have seen R built-in functions to compute permutations and combinations. For a better understanding, you are now required to create your own functions for

1. Permutations with replacement
2. Permutations without replacement
3. Combinations with replacement
4. Combinations without replacement

you are going to write 4 independent functions taking 2 parameters n and r and return the number of ways to choose r from n using the 4 methods discussed above.

Hints

you can use factorial built in functions : factorial()

$$P(n,k) = \frac{n!}{(n-k)!}.$$

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$